Singularity In A Nutshell

Galen C. Hunt, Ph.D.
Principal Researcher, OS Group
Microsoft Corporation

Singularity

Building dependable software systems

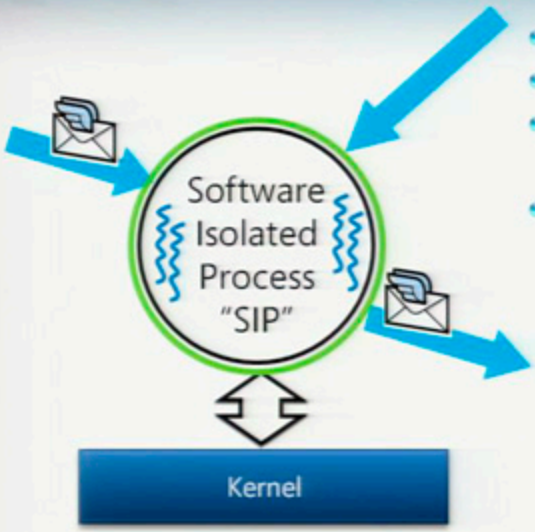
- Research OS
  - not a product
  - not a replacement for Windows
  - test bed for new ideas
- Goal is to
  - overcome faults and limits of current OS's
  - drastically improve system dependability and security
  - achieve good enough performance

Architecture
Founded on static verification

- Singularity contributes three key ideas:
  - Software-Isolated Processes (SIPs)
  - Contract-Based Channels (CBCs)
  - Manifest-Based Programs (MBPs)

Software-Isolated Processes

Verified process protection



- Set of code and data pages.
- Data pages contain typed objs
- Code pages contain safe code
  - (type safe and memory safe)
- => No need for HW protection

Contract-Based Channels

Verified inter-process communciation

- Contracts define messages and exchange patterns for inter-process communication.

```
contract TcpConnectionContract {
  ...
  state Connected : one {
    Read? -> Reading;
    Write? -> Writing;
    ...
  }

  state Reading : one {
    Data! -> Connected;
    NoMoreData! -> SendOnly;
    RemoteClose! -> Zombie;
  }
  ...
}
```

Contract-Based Channels

Verified inter-process communciation

- Contracts define messages and exchange patterns for inter-process communication.

```
contract TcpConnectionContract {
  ...
  state Connected : one {
    Read? -> Reading;
    Write? -> Writing;
    ...
  }

  state Reading : one {
    Data! -> Connected;
    NoMoreData! -> SendOnly;
    RemoteClose! -> Zombie;
  }
  ...
}
```

```
...
conn.SendRead();
switch receive {
  case conn.Data(readData) :
    dataBuffer.AddToTail(readData);
    return true;

  case conn.RemoteClose() :
    return false;
}
...
```

Contract-Based Channels

Verified inter-process communciation

- Contracts define messages and exchange patterns for inter-process communication.

```
contract TcpConnectionContract {
  ...
  state Connected : one {
    Read? -> Reading;
    Write? -> Writing;
    ...
  }

  state Reading : one {
    Data! -> Connected;
    NoMoreData! -> SendOnly;
    RemoteClose! -> Zombie;
  }
  ...
}
```

```
...
conn.SendRead();
switch receive {
  case conn.Data(readData) :
    dataBuffer.AddToTail(readData);
    return true;

  case conn.RemoteClose() :
    return false;
}
...
```

Missing Case

```
case conn.NoMoreData() :
```

Contract-Based Channels

Verified inter-process communciation

- Contracts define messages and exchange patterns for inter-process communication.

```
contract TcpConnectionContract {
  ...
  state Connected : one {
    Read? -> Reading;
    Write? -> Writing;
    ...
  }

  state Reading : one {
    Data! -> Connected;
    NoMoreData! -> SendOnly;
    RemoteClose! -> Zombie;
  }
  ...
}
```

```
...
conn.SendRead();
switch receive {
  case conn.Data(readData) :
    dataBuffer.AddToTail(readData);
    return true;

  case conn.RemoteClose() :
    return false;
}
...
```

Missing Case

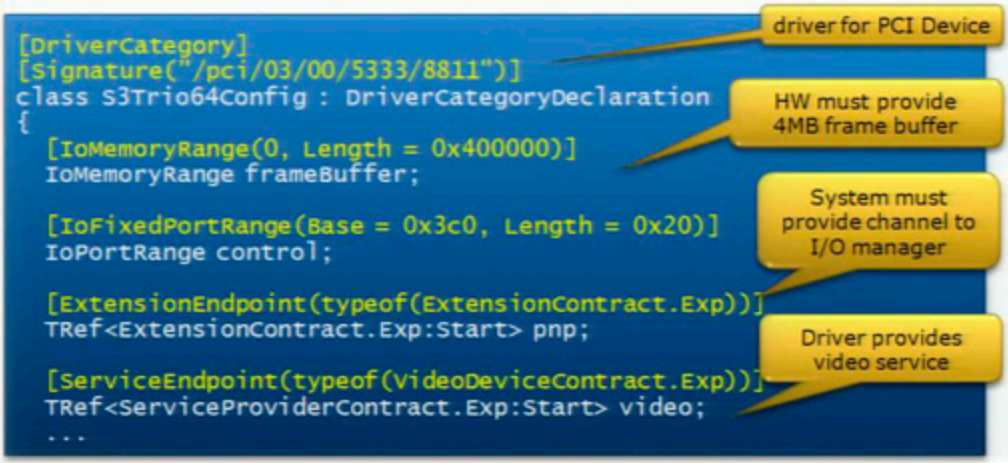
```
case conn.NoMoreData() :
```

Contract conformance statically detects subtle errors.

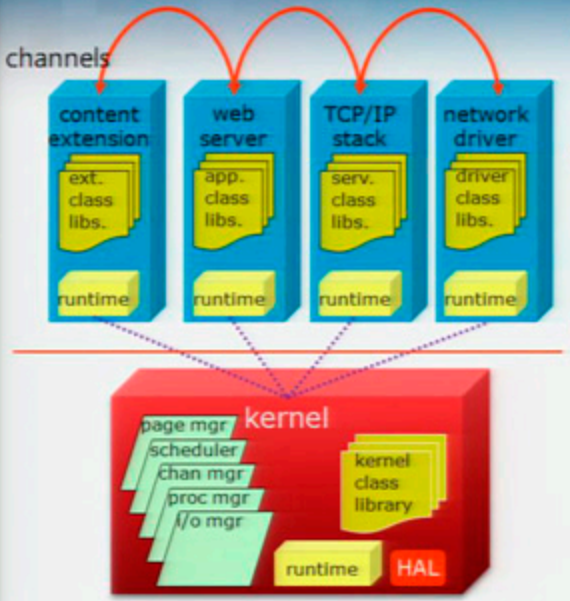
Manifest-Based Programs

Verified system composition

- Programs declare requirements and services through manifests



Singularity Version 1.0



- Micro-kernel like design
  - Services and Drivers in SIPs
- Safe Kernel and System
  - 95% written in C#
  - 17% of files contain unsafe C#
  - 5% of files contain x86 or C++
- "Complete" Server OS
  - Runs SPECweb99, etc.
  - TCP/IP stack, file system, etc.
  - Wide classes of device drivers

Singularity Is A Research Vehicle

- We wrote Singularity with the following goal:
  - Imagine you have a new OS research idea, you can produce a strong publishable result in 1 or 2 man-years
- Architecture is clean
  - Simple, orthogonal abstractions
  - Simple implementation

Ongoing Research In MSR
Singularity version 2.0

- A few areas of active research:
  - Compile-Time Reflection
  - Hardware Protection Domains
  - Heterogeneous Many-core Processing
  - Type Assembly Language
  - Post-PC Computing (cPhone)

announcing

Singularity Version 1.0
Research Development Kit

Summary

- Singularity is basis for more dependable systems
  - pervasive use of safe programming languages
  - lightweight, closed, customizable run-time environment
  - verifiable specification of system behavior
- Working research prototype
  - driving research in large number of areas
- For more information:
  - http://research.microsoft.com/os/singularity
  - Start with the *Operating Systems Review* paper.
  - RFP: mark.lewin@microsoft.com

Martin Abadi, Mark Aiken, Paul Barham, Andrew Birrell, Richard Black, Chris Brumme, Kent Cedola, Juan Chen, Trishul Chilimbi, Ken Church, Arlie Davis, John DeTreville, Austin Donnelly, Rob Earhart, Raymond Endres, Ulfar Erlingsson, Manuel Fahndrich, Bruno Garagnon, [illegible] Grieskamp, Tim Harris, Chris Hawb[illegible]n, Galen Hunt, Rebecca Isaacs, Mike Jones, Ashok Kuppusamy, Jim Larus, Steven Levi, Roy Levin, Qunyan Mangus, Virendra Marathe, Adrian Marinescu, Nick Murphy, Dushyanth Narayanan, Kirk Olynyk, Ravi Pandya, Stathis Papaefstathiou, Mark Plesko, Sriram Rajamani, Jakob Rehof, John Richardson, Evan Schrier, Wolfram Schulte, Valerie See, Dan Simon, David Stahlkopf, Bjarne Steensgaard, Aaron Stern, Clemens Szyperski, David Tarditi, Chandu Thekkath, Dean Tribble, Sean Trowbridge, Ted Wobber, Songtao Xia, Brian Zill, Ben Zorn, Martin Abadi, Chris Brumme, Kent Cedola, Juan

Microsoft®

Your potential. Our passion.™